**Duration**: 3 Hours  
**Total Marks**: 100  
**Instructions**: Attempt all sections. This paper is designed to assess deep practical knowledge and problem-solving in frontend web technologies.

**Theory (40 Marks)**

**Q1. HTML & Accessibility (10 Marks)**

a) What is ARIA in HTML? Explain how it helps in accessibility with 2 examples. (5 Marks)  
b) Create a semantic HTML structure for a blog post that includes a title, image, author name, publish date, content, tags, and comment section. Use only semantic elements. (5 Marks)

**Q2. CSS & Layout Mastery (10 Marks)**

a) Explain how flex and grid systems differ with at least two real-world use cases for each. (5 Marks)  
b) Write a CSS-only solution for a tooltip that appears above an element on hover, stays for 3 seconds, and fades out smoothly. No JS allowed. (5 Marks)

**Q3. JavaScript Core (10 Marks)**

a) Deep dive: What happens behind the scenes when setTimeout is called in JS? Explain in terms of the event loop, callback queue, and call stack. (5 Marks)  
b) Write a function to **debounce** another function in JavaScript. Explain how it helps performance in real-world apps. (5 Marks)

**Q4. jQuery + Bootstrap Theory (10 Marks)**

a) What’s the difference between .on() and .click() in jQuery in terms of event delegation and dynamic content? Explain with code. (5 Marks)  
b) Explain how Bootstrap 5 handles responsiveness differently from Bootstrap 3/4. Talk about the container system, breakpoints, and utility classes. (5 Marks)

**Practical Implementation (60 Marks)**

**Q5. Component Challenge (15 Marks)**

Build a **responsive image gallery** using Bootstrap 5, where:

* Images are shown in a grid layout
* Clicking an image opens a modal with the full-size image and caption
* The layout shifts from 4-column to 2-column to 1-column as screen size reduces
* Add a dark mode toggle with JavaScript

**Q6. JavaScript Logic & DOM (15 Marks)**

Build a **custom dropdown component** using vanilla JavaScript:

* No <select> allowed
* Should support keyboard navigation (↑ ↓ Enter)
* Clicking outside should close it
* Support for disabling certain items in the list
* Add styles via CSS to mimic Bootstrap’s dropdown (no actual Bootstrap usage)

**Q7. jQuery Animation + API (15 Marks)**

Create a **quote viewer** using jQuery:

* Fetch a random quote from an open API like <https://api.quotable.io/random>
* Animate the quote in with a fade/slide
* Show a loading spinner while fetching
* Include a “New Quote” button
* Style with Bootstrap

**Q8. CSS Wizardry (15 Marks)**

Write **pure CSS** to draw the following:

* A button with a glowing border effect on hover
* A loading spinner with animation (no images)
* A responsive "card" layout with hover flip animation (front & back content)

Bonus: Add a theme switcher (dark/light) using only CSS and input checkbox hack.

**Q9. Code Debugging (5 Marks)**

You are given the following HTML and JS. Fix the bugs:

<button class="btn">Click</button>

<script>

document.querySelectorAll(".btn").forEach(btn => {

btn.addEventListener("onclick", () => {

alert("Button clicked");

});

});

</script>

What’s wrong? Explain and rewrite correctly.

**Q10. Real-world Thinking (5 Marks)**

Imagine you're building a dashboard with heavy DOM manipulation and frequent API calls.  
a) Which practices would you follow to keep the frontend fast and responsive? (Any 3)  
b) How would you structure your CSS and JS files for scalability in a large project?